(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 107463402 A
(43)申请公布日 2017.12.12

(21)申请号 201710640433.7

(22)申请日 2017.07.31

(71)申请人 腾讯科技(深圳)有限公司
地址 518000 广东省深圳市南山区高新区科技中一路腾讯大厦35层

(72)发明人 庄志伟

(74)专利代理机构 北京康信知识产权代理有限责任公司 11240
代理人 赵囡囡 褚敏

(51)Int.Cl.
G06F 9/445(2006.01)
G06F 9/455(2006.01)

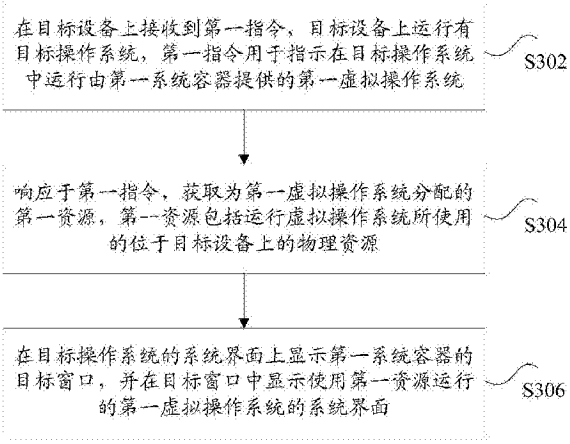
权利要求书3页 说明书14页 附图5页

(54)发明名称

虚拟操作系统的运行方法和装置

(57)摘要

本发明公开了一种虚拟操作系统的运行方法和装置。其中，该方法包括：在目标设备上接收到第一指令，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统；响应于第一指令，获取为第一虚拟操作系统分配的第一资源，第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源；在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面，第一虚拟操作系统的系统内核为目标操作系统的系统内核，第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。本发明解决了相关技术中虚拟机的启动较慢的技术问题。



CN 107463402 A

1.一种虚拟操作系统的运行方法，其特征在于，包括：

在目标设备上接收到第一指令，其中，所述目标设备上运行有目标操作系统，所述第一指令用于指示在所述目标操作系统中运行由第一系统容器提供的第一虚拟操作系统；

响应于所述第一指令，获取为所述第一虚拟操作系统分配的第一资源，其中，所述第一资源包括运行所述虚拟操作系统所使用的位于所述目标设备上的物理资源；

在所述目标操作系统的系统界面上显示所述第一系统容器的目标窗口，并在所述目标窗口中显示使用所述第一资源运行的所述第一虚拟操作系统的系统界面，其中，所述第一虚拟操作系统的系统内核为所述目标操作系统的系统内核，所述第一虚拟操作系统的系统驱动为所述目标操作系统的系统驱动。

2.根据权利要求1所述的方法，其特征在于，获取为所述第一虚拟操作系统分配的第一资源包括：

获取由所述目标操作系统的系统内核为所述第一虚拟操作系统分配的所述第一资源，其中，所述第一资源与由所述目标操作系统的系统内核为第二虚拟操作系统分配的第二资源不同，在所述目标操作系统中还运行有由第二系统容器提供的所述第二虚拟操作系统，所述第二资源包括运行所述第二虚拟操作系统所使用的所述目标设备上的物理资源。

3.根据权利要求1或2所述的方法，其特征在于，在使用所述第一资源运行所述第一虚拟操作系统的过程中，所述方法还包括：

将第一文件目录作为所述第一虚拟操作系统的根目录，其中，所述第一文件目录为所述目标操作系统的系统内核预先为所述第一虚拟操作系统分配的，所述第一虚拟操作系统的根目录用于保存所述第一虚拟操作系统运行所需的文件；

在所述第一资源上运行第一进程，其中，所述第一进程为所述第一虚拟操作系统的系统进程，所述第一进程用于对所述第一文件目录下的文件进行处理，所述第一进程不同于第二进程，所述第二进程为第二虚拟操作系统的系统进程，在所述目标操作系统中还运行有由第二系统容器提供的所述第二虚拟操作系统。

4.根据权利要求1或2所述的方法，其特征在于，在接收到所述第一指令之前，在所述目标操作系统的系统内核不包括目标工具的情况下，所述方法还包括：

在接收到第二指令时，按照所述第二指令的指示将所述目标工具安装至所述目标操作系统的系统内核中，其中，添加至所述目标操作系统的系统内核中的所述目标工具用于为不同的系统容器分配相互隔离的物理资源。

5.根据权利要求1或2所述的方法，其特征在于，在接收到所述第一指令之前，在所述目标操作系统的系统内核不包括指令集合的情况下，所述方法还包括：

在接收到第三指令时，按照所述第三指令的指示将所述指令集合添加至所述目标操作系统中，其中，添加至所述目标操作系统中的所述指令集合用于提供运用所述目标操作系统中的目标工具所需的指令。

6.根据权利要求5所述的方法，其特征在于，在按照所述第三指令的指示将所述指令集合添加至所述目标操作系统中之后，所述方法还包括：

在接收到第四指令时，按照所述第四指令的指示运行所述目标工具，以创建所述第一系统容器和所述第一系统容器的配置文件，其中，所述指令集合包括所述第四指令，所述配置文件用于配置为所述第一系统容器分配的物理资源和为所述第一系统容器分配的文件

目录。

7.根据权利要求1或2所述的方法，其特征在于，所述目标操作系统的系统内核为Linux内核，所述第一虚拟操作系统为基于Linux内核实现的操作系统。

8.一种虚拟操作系统的运行装置，其特征在于，包括：

接收单元，用于在目标设备上接收到第一指令，其中，所述目标设备上运行有目标操作系统，所述第一指令用于指示在所述目标操作系统中运行由第一系统容器提供的第一虚拟操作系统；

响应单元，用于响应于所述第一指令，获取为所述第一虚拟操作系统分配的第一资源，其中，所述第一资源包括运行所述虚拟操作系统所使用的位于所述目标设备上的物理资源；

显示单元，用于在所述目标操作系统的系统界面上显示所述第一系统容器的目标窗口，并在所述目标窗口中显示使用所述第一资源运行的所述第一虚拟操作系统的系统界面，其中，所述第一虚拟操作系统的系统内核为所述目标操作系统的系统内核，所述第一虚拟操作系统的系统驱动为所述目标操作系统的系统驱动。

9.根据权利要求8所述的装置，其特征在于，所述响应单元还用于获取由所述目标操作系统的系统内核为所述第一虚拟操作系统分配的所述第一资源，其中，所述第一资源与由所述目标操作系统的系统内核为第二虚拟操作系统分配的第二资源不同，在所述目标操作系统中还运行有由第二系统容器提供的所述第二虚拟操作系统，所述第二资源包括运行所述第二虚拟操作系统所使用的所述目标设备上的物理资源。

10.根据权利要求8或9所述的装置，其特征在于，所述装置还包括：

第一处理单元，用于在使用所述第一资源运行所述第一虚拟操作系统的过程中，将第一文件目录作为所述第一虚拟操作系统的根目录，其中，所述第一文件目录为所述目标操作系统的系统内核预先为所述第一虚拟操作系统分配的，所述第一虚拟操作系统的根目录用于保存所述第一虚拟操作系统运行所需的文件；

运行单元，用于在所述第一资源上运行第一进程，其中，所述第一进程为所述第一虚拟操作系统的系统进程，所述第一进程用于对所述第一文件目录下的文件进行处理，所述第一进程不同于第二进程，所述第二进程为第二虚拟操作系统的系统进程，在所述目标操作系统中还运行有由第二系统容器提供的所述第二虚拟操作系统。

11.根据权利要求8或9所述的装置，其特征在于，所述装置还包括：

第二处理单元，用于在所述目标操作系统的系统内核不包括目标工具的情况下，在接收到第二指令时，按照所述第二指令的指示将所述目标工具安装至所述目标操作系统的系统内核中，其中，添加至所述目标操作系统的系统内核中的所述目标工具用于为不同的系统容器分配相互隔离的物理资源。

12.根据权利要求8或9所述的装置，其特征在于，所述装置还包括：

第三处理单元，用于在所述目标操作系统的系统内核不包括指令集合的情况下，在接收到第三指令时，按照所述第三指令的指示将所述指令集合添加至所述目标操作系统中，其中，添加至所述目标操作系统中的所述指令集合用于提供运用所述目标操作系统中的目标工具所需的指令。

13.根据权利要求12所述的装置，其特征在于，所述装置还包括：

创建单元，用于在按照所述第三指令的指示将所述指令集合添加至所述目标操作系统中之后，在接收到第四指令时，按照所述第四指令的指示运行所述目标工具，以创建所述第一系统容器和所述第一系统容器的配置文件，其中，所述指令集合包括所述第四指令，所述配置文件用于配置为所述第一系统容器分配的物理资源和为所述第一系统容器分配的文件目录。

14.一种存储介质，其特征在于，所述存储介质包括存储的程序，其中，所述程序运行时执行所述权利要求1至7中任一项所述的方法。

15.一种电子装置，包括存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，其特征在于，所述处理器通过所述计算机程序执行所述权利要求1至7任一项中所述的方法。

虚拟操作系统的运行方法和装置

技术领域

[0001] 本发明涉及互联网领域，具体而言，涉及一种虚拟操作系统的运行方法和装置。

背景技术

[0002] 虚拟化，是指通过虚拟化技术将一台计算机虚拟为多台逻辑计算机。在一台计算机上同时运行多个逻辑计算机，每个逻辑计算机可运行不同的操作系统，并且应用程序都可以在相互独立的空间内运行而互不影响，从而显著提高计算机的工作效率。虚拟化使用软件的方法重新定义划分IT资源，可以实现IT资源的动态分配、灵活调度、跨域共享，提高IT资源利用率，使IT资源能够真正成为社会基础设施，服务于各行各业中灵活多变的应用需求。

[0003] 在相关技术中，虚拟化的实现方式如图1所示，首先在Host主机的硬件和软件资源(如主机自带的系统资源、硬件资源等)上使用QEMU来虚拟出不同的CPU(如X86、PPC、SPARC)等硬件设备，然后在QEMU虚拟的不同硬件设备上使用不同的驱动Drivers和不同的系统(如Linux、Windows等)，然后用户使用例如KVM、VMWare这样的管理软件，在虚拟出的系统中运行用户自己的应用程序、操作文件(安装或保存在用户空间内)。

[0004] 在相关技术中，存在如下问题：系统启动慢，需要消耗较长时间。

[0005] 因为相关技术的方案中，在启动系统前需要先用QEMU虚拟CPU等硬件设备，然后才能启动系统。而整个QEMU虚拟过程需要虚拟物理层CPU、磁盘、图形处理设备、网络设备等设备。在整个虚拟硬件设备的过程中，需要使用CPU提供的虚拟化支持VT-X(是intel运用Virtualization虚拟化技术中的一个指令集)为Guest OS(即客机或虚拟机)创建虚拟化处理器，使用kvm(是Kernel-based Virtual Machine的简称，是一个开源的系统虚拟化模块)将主机内存虚拟化成独立的虚拟化内存的地址，IO请求被QEMU截取，以完成对硬件设备的虚拟化，整个过程较之直接使用Host主机的物理设备而言，显得极其缓慢，拖慢了系统的启动时间。

[0006] 且由于现有系统需要虚拟出不同的硬件设备，所以虚拟出的硬件在运行对应的虚拟操作系统时，还需启动每个虚拟操作系统中自己的一套驱动程序，进一步降低了启动的效率。

[0007] 针对相关技术中虚拟机的启动较慢的技术问题，目前尚未提出有效的解决方案。

发明内容

[0008] 本发明实施例提供了一种虚拟操作系统的运行方法和装置，以至少解决相关技术中虚拟机的启动较慢的技术问题。

[0009] 根据本发明实施例的一个方面，提供了一种虚拟操作系统的运行方法，该运行方法包括：在目标设备上接收到第一指令，其中，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统；响应于第一指令，获取为第一虚拟操作系统分配的第一资源，其中，第一资源包括运行虚拟操作系统所使

用的位于目标设备上的物理资源;在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面,其中,第一虚拟操作系统的系统内核为目标操作系统的系统内核,第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0010] 根据本发明实施例的另一方面,还提供了一种虚拟操作系统的运行装置,该运行装置包括:接收单元,用于在目标设备上接收到第一指令,其中,目标设备上运行有目标操作系统,第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统;响应单元,用于响应于第一指令,获取为第一虚拟操作系统分配的第一资源,其中,第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源;显示单元,用于在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面,其中,第一虚拟操作系统的系统内核为目标操作系统的系统内核,第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0011] 在本发明实施例中,在目标设备上接收到第一指令时,获取为第一虚拟操作系统分配的第一资源,在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统(即虚拟机)的系统界面,第一虚拟操作系统直接调用目标操作系统的系统内核,并直接调用目标操作系统的系统驱动来运行,可以解决了相关技术中虚拟机的启动较慢的技术问题,进而达到提高虚拟机的启动速度的技术效果。

附图说明

[0012] 此处所说明的附图用来提供对本发明的进一步理解,构成本申请的一部分,本发明的示意性实施例及其说明用于解释本发明,并不构成对本发明的不当限定。在附图中:

[0013] 图1是相关技术中的一种可选的虚拟机系统的示意图;

[0014] 图2是根据本发明实施例的虚拟操作系统的运行方法的硬件环境的示意图;

[0015] 图3是根据本发明实施例的一种可选的虚拟操作系统的运行方法的流程图;

[0016] 图4是根据本发明实施例的一种可选的虚拟机系统的示意图;

[0017] 图5是根据本发明实施例的一种可选的内核编译选项的示意图;

[0018] 图6是根据本发明实施例的一种可选的虚拟操作系统的运行装置的示意图;

[0019] 图7是根据本发明实施例的一种可选的虚拟操作系统的运行装置的示意图;以及

[0020] 图8是根据本发明实施例的一种终端的结构框图。

具体实施方式

[0021] 为了使本技术领域的人员更好地理解本发明方案,下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本发明一部分的实施例,而不是全部的实施例。基于本发明中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都应当属于本发明保护的范围。

[0022] 需要说明的是,本发明的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象,而不必用于描述特定的顺序或先后次序。应该理解这样使用

的数据在适当情况下可以互换，以便这里描述的本发明的实施例能够以除了在这里图示或描述的那些以外的顺序实施。此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0023] 首先，在对本发明实施例进行描述的过程中出现的部分名词或者术语适用于如下解释：

[0024] CGroup:Cgroups是control groups的缩写，是Linux内核提供的一种可以限制、记录、隔离进程组(process groups)所使用的物理资源(如:CPU、memory、IO等等)的机制。

[0025] NameSpace:NameSpace是Linux内核中用来实现软件资源隔离的机制，隔离处理主机名与域名、挂载点(文件资源)、进程编号、信息量、信息队列、共享内存、网络设备、端口网络栈等虚拟化需要创建的软件资源。

[0026] 虚拟化:是指通过虚拟化技术将一台计算机虚拟为多台逻辑计算机。在一台计算机上同时运行多个逻辑计算机，每个逻辑计算机可运行不同的操作系统，并且应用程序都可以在相互独立的空间内运行而互不影响，从而显著提高计算机的工作效率。具体可在一个物理设备中使用修改的操作系统代码和驱动，实现虚拟出单个或者多个硬件资源共享软件资源，并相互隔离和相互切换的操作系统的技术。

[0027] LXC容器:LXC为Linux Container的简写。可以提供轻量级的虚拟化，以便隔离进程和资源，而且不需要提供指令解释机制以及全虚拟化的其他复杂性。相当于C++中的NameSpace。容器有效地将由单个操作系统管理的资源划分到孤立的组中，以更好地在孤立的组之间平衡有冲突的资源使用需求。

[0028] Host:为相关虚拟容器提供物理资源和软件资源的android设备或者是虚拟android设备。

[0029] QEMU:是一套编写的以GPL许可证分发源码的模拟处理器，在GNU/Linux平台上使用广泛。

[0030] busybox:是一个集成了一百多个最常用Linux命令和工具的软件。

[0031] PID:全称为Process Identification，操作系统里指进程识别号，也就是进程标识符。操作系统里每打开一个程序都会创建一个进程ID，即PID。

[0032] Adb:全称为Android Debug Bridge，起到调试桥的作用。借助adb工具，可以管理设备或手机模拟器的状态，还可以进行很多手机操作，如安装软件、系统升级、运行shell命令等等。

[0033] 实施例1

[0034] 根据本发明实施例，提供了一种虚拟操作系统的运行方法的方法实施例。

[0035] 可选地，在本实施例中，上述虚拟操作系统的运行方法可以应用于如图2所示的由服务器202和终端204所构成的硬件环境中。如图2所示，服务器202通过网络与终端204进行连接，上述网络包括但不限于:广域网、城域网或局域网，终端204并不限定于PC、手机、平板电脑等。本发明实施例的虚拟操作系统的运行方法可以由服务器202来执行，也可以由终端204来执行，还可以是由服务器202和终端204共同执行。其中，终端204执行本发明实施例的虚拟操作系统的运行方法也可以是由安装在其上的客户端来执行。

[0036] 图3是根据本发明实施例的一种可选的虚拟操作系统的运行方法的流程图，如图3所示，该方法可以包括以下步骤：

[0037] 步骤S302，在目标设备上接收到第一指令，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统。

[0038] 上述的目标设备运行以Linux内核(或Unix内核)为内核的系统(也即目标操作系统)，所运行的系统包括但不局限于Linux、Ubuntu、Android，该目标设备的类型包括但不局限于移动设备、计算机终端PC、服务器；目标操作系统中安装有一个或多个系统容器，每个系统容器用于提供一个虚拟操作系统，任意两个系统容器所提供的虚拟操作系统可以相同或者不同。

[0039] 在目标设备为移动设备或PC时，通过在移动设备或PC中提供系统容器(相当于虚拟操作系统)，方便用户以安全模式使用移动设备或PC，即每个用户仅能进入相应的系统容器所提供的虚拟操作系统，且多个用户所使用的系统容器相互独立、物理隔离，不会相互影响，可以提高移动设备或PC中信息的安全性。

[0040] 在目标设备为服务器时，服务器可以通过云服务的形式提供若干个系统容器，每个用户可通过互联网使用自己的系统容器，一方面可以降低用户本地终端成本(仅需一个联网设备即能进入到自己的虚拟系统)，而不用频繁升级设备；另一方面可以提高信息的安全性(由于所使用的系统容器相互独立、物理隔离)。

[0041] 步骤S304，响应于第一指令，获取为第一虚拟操作系统分配的第一资源，第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源。

[0042] 相关技术中，对于不同的虚拟操作系统，存在共用CPU、内存等物理资源的情况，从而存在其中一个虚拟操作系统运行时的信息被其它操作系统知晓(如存在于共享内存中的信息)的可能性，从而影响了虚拟操作系统的安全性。

[0043] 对于本申请的任意两个系统容器(相当于两个虚拟操作系统)，所分配的用于运行或承载系统容器的物理资源相互独立隔离，如所使用的CPU资源、内存资源等不相同，从而可以避免虚拟操作系统运行过程中的信息被其它虚拟操作系统知晓，提高虚拟操作系统运行的安全性。

[0044] 步骤S306，在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面，第一虚拟操作系统的系统内核为目标操作系统的系统内核，第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0045] 在相关技术中，系统启动慢，需要消耗较长时间的原因主要包括如下两个方面：(1)在启动系统前需要先用QEMU虚拟CPU等硬件设备，然后才能启动系统；(2)驱动独立且无法共享，现有系统需要虚拟出不同的硬件设备，即使虚拟出相同的硬件，每个虚拟操作系统启动时都需要运行自己的一套驱动程序。

[0046] 另外，在相关技术中，还存在技术成本高、误码率高、传输速度慢、安全性差等类似问题，对于个人用户体验不佳等等缺陷。

[0047] 对于在目标操作系统中运行的虚拟操作系统，直接利用目标操作系统中的系统内核和系统驱动进行运行，而不用为每个虚拟操作系统虚拟出物理层CPU、磁盘、图形处理设备、网络设备、内存、驱动、系统内核等，在虚拟操作系统启动时仅需建立相应的系统进程，

初始化相关目录(如根目录),并调用目标操作系统中的系统内核和系统驱动进行运行即可,而不用启动虚拟出的物理层CPU、磁盘、图形处理设备、网络设备、内存、驱动、系统内核等。从而可以解决相关技术中的问题,达到提高虚拟系统的启动效率的效果。

[0048] 可选地,对于每一个系统容器,可归属于一个用户(如以云服务的形式提供给用户使用),当用户需要对系统容器提供的虚拟操作系统进行升级时,可以根据用户指示的系统版本对系统容器提供的虚拟操作系统进行升级。

[0049] 采用上述升级方式好处在于,用户可以远程登录该终端,而不用升级本地设备,对于用户没有任何影响。且能够降低用户升级设备(如手机、平板)的消费,同时通过最新的系统带给用户最好的体验。

[0050] 通过上述步骤S302至步骤S306,在目标设备上接收到第一指令时,获取为第一虚拟操作系统分配的第一资源,在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统(即虚拟机)的系统界面,第一虚拟操作系统直接调用目标操作系统的系统内核,并直接调用目标操作系统的系统驱动来运行,可以解决了相关技术中虚拟机的启动较慢的技术问题,进而达到提高虚拟机的启动速度的技术效果。

[0051] (一)创建系统容器及其运行环境

[0052] 本申请使用了Host主机(即目标设备)的Linux系统内核,使用Host的Drivers驱动,为了使得主机驱动的兼容性更好,可以在驱动中进行部分代码修改,使用LXC工具(即目标工具)作为虚拟化容器的启动和管理工具,来在Host上启动多个相互隔离的容器,以提供给用户使用,为了便于描述,后续的目标设备以Android主机为例进行说明,如图4所示:

[0053] 在修改过的Host Android主机上修改Android使用的内核版本,使用Linux内核的CGroups和NameSpace功能来支持硬件的虚拟化和软件资源的虚拟化。使用NameSpace和PID完成软件资源的隔离,并使用LXC工具(LXC Tools)为每个容器启动一个启动进程并创建NameSpace(即命名空间)和对应的CGroup(分组化管理的进程),并且依据LXC容器(也即系统容器)中的配置文件(即config文件)中的参数进行初始化配置(如配置相关的虚拟硬件参数),然后在软件和硬件资源准备完成后,进程调用“init.rc”(系统初始化配置文件)来启动初始化和启动虚拟的操作系统。

[0054] 从上述内容可以看出,在创建系统容器及其运行环境的过程中,主要包括以下几个部分:

[0055] 1)对Host Android系统的修改,需要对Android源码和内核的代码进行修改;

[0056] 2)LXC工具需要通过交叉编译移植到Android系统,使LXC工具可以在Android系统中运行;

[0057] 3)为Host Android系统添加busybox工具,添加Android系统使用LXC工具缺失的指令和应用;

[0058] 4)提取和制作可以在Android系统中使用的LXC容器(包括系统启动运行需要的相关文件和LXC的配置文件);

[0059] 5)修改添加内核驱动代码和相关管理代码用于切换虚拟的操作系统。

[0060] (1)关于对Host Android系统的修改

[0061] 为了在Host Android系统中使用LXC tools,需要系统内核支持CGroup和

NameSpace,同时需要修改Android源码去除掉对PIE限制。

[0062] 为了启动Android Linux内核的CGroup和NameSpace功能,需要修改Android linux内核的编译配置,一种可选的新加配置选项以下代码所示:

[0063] 如对于NameSpace功能:

[0064] CONFIG_NAMESPACES=y

[0065] CONFIG_IPC_NS=y

[0066] 上述“CONFIG_NAMESPACES=y”相当于开启NameSpace功能,“CONFIG_IPC_NS=y”相当于开启IPC功能。

[0067] 对于CGroup功能:

[0068] CONFIG_CGROUPS=y

[0069] CONFIG_CPUSETS=y

[0070] 上述“CONFIG_CGROUPS=y”相当于开启CGroup功能,“CONFIG_CPUSETS=y”相当于允许设置CPU。

[0071] 可选地,为了使得Linux内核的运行更加稳定,需要根据Android Linux内核的版本选择匹配的LXC工具的版本。

[0072] 在修改Android源码以去除对PIE的限制时,还需要修改“linker.cpp”(用于执行Android的动态库的链接工作),注释掉或者删除与PIE相关的代码。

[0073] (2)关于LXC工具的交叉编译移植

[0074] 在目标操作系统的系统内核不包括目标工具的情况下,在接收到第二指令(也即对LXC工具进行交叉编译和移植的指令)时,按照第二指令的指示将目标工具通过交叉编译后安装至目标操作系统中,添加至目标操作系统中的目标工具用于为不同的系统容器分配相互隔离的物理资源。

[0075] 为了实现对LXC工具的交叉编译,可使用“android ndk”这一交叉编译工具,配置对应的交叉编译参数arch和对应的编译器gcc的配置,同时还需要编译libcap(一种Linux环境下的数据包捕获开发包),添加和替换掉LXC工具中的相关文件(替换添加到“lxc/lib”中,因为lxc默认是在Ubuntu系统下运行而不是为移动版本的linux环境编写的,所以将lxc移植过程中需要添加移动版linux中lxc使用,但是没有的libcap库,所以需要添加替换)。因为Android系统中的配置文件不是config文化,而是一个“config.gz”的压缩文件包,因此在交叉编译成功后修改LXC工具中对系统内核支持的检测脚本。

[0076] (3)关于添加busybox工具

[0077] 在目标操作系统的系统内核不包括指令集合的情况下,可在接收到第三指令(该指令即添加busybox工具的指令)时,按照第三指令的指示将指令集合添加至目标操作系统中,添加至目标操作系统中的指令集合用于提供运用目标操作系统中的目标工具所需的指令。

[0078] 上述的指令集合即由busybox工具提供的。

[0079] 对于Android主机,可下载对应版本的busybox工具,使用adb工具链接到Host Android系统,并将busybox存放入至Host Android的存储空间中然后解压安装busybox,以添加Host Android系统缺失的LXC工具运行需要的指令。

[0080] (4)关于提取和制作Android系统容器

[0081] 在按照第三指令的指示将指令集合添加至目标操作系统中之后，在接收到第四指令(即用于提取或制作Android系统容器的指令)时，按照第四指令的指示运行目标工具，以创建第一系统容器和第一系统容器的配置文件，指令集合包括第四指令，配置文件用于配置为第一系统容器分配的物理资源和为第一系统容器分配的文件目录。

[0082] 上述的LXC操作系统容器包括1个名为“rootfs”的文件夹(如第一文件目录)和一个与LXC相关的config配置文件，rootfs文件夹包括系统启动需要的相关文件和目录，当然如果使用Android系统部分目录可以使用挂载的方式共享使用Host Android系统中的相关目录，config配置文件需要配置“lxc.net”等LXC的相关配置，rootfs文件夹目录可以从android系统中使用adb指令提取和压缩对应系统文件生成，也可以使用config配置中添加LXC工具虚拟化容器的启动过程Hook脚本，在LXC工具启动前调用Hook脚本挂载Host Android系统完成rootfs的相关目录和文件的生成。

[0083] (5)关于修改添加内核驱动代码

[0084] 可修改“drivers/staging/android/.drivers/rtc/”和“drivers/staging/android/”等目录下的文件，以添加与源码修改对应的编译配置，从而完成新的驱动编译。

[0085] 图5为修改内核编译选项和Host Android源码后交叉编译LXC工具后将LXC工具放入Host Android存储空间，为Host Android安装完成busybox指令支持后，运行的修改后的LXC工具内核支持检测脚本的运行结果。

[0086] 使用的检测脚本式lxc项目编译完成后自带的lxc-checkconfig文件，运行这个脚本会查找系统中前文提到的config.gz文件，并检测其中修改的那部分内核配置代码是否有效，从而检测lxc所需的虚拟化运行环境。如关于Namespace的状态，Namespace功能、PID功能、User namespace(即用户控制)被启用(即enabled)；关于Cgroup的状态，Cgroup功能、Cgroup device(与Cgroup相关的设备)、Cgroup CPU account(CPU控制帐号)功能被启用。

[0087] 在本申请的技术方案中，提出了一种轻量级的android系统虚拟化方法，可以用于物理android机器的多个android系统的虚拟化，提高虚拟化效率，最大限度的提升物理硬件的使用率，也可以用于虚拟android系统中虚拟出多个轻量级别的虚拟系统，使虚拟设备数量得到大量提升，用于服务器上的大量android系统的虚拟需求。

[0088] (二)运行系统容器

[0089] 在步骤S304提供的技术方案中，获取为第一虚拟操作系统分配的第一资源包括：获取由目标操作系统的系统内核为第一虚拟操作系统分配的第一资源，第一资源与由目标操作系统的系统内核为第二虚拟操作系统分配的第二资源不同，在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统，第二资源包括运行第二虚拟操作系统所使用的目标设备上的物理资源。

[0090] LXC操作系统容器包括1个名为“rootfs”的文件夹(如第一文件目录)和一个与LXC相关的config配置文件，可根据配置文件确定系统内核中的目标工具(如LXC)为第一虚拟操作系统分配的内存、CPU、GPU等资源，并利用这些资源来运行第一虚拟操作系统(如虚拟Android操作系统)，在为不同的系统容器分配硬件资源时，分配的资源不同，以保证系统之间具有良好的隔离性，保证系统运行的安全性。

[0091] 在步骤S306提供的技术方案中，在使用第一资源运行第一虚拟操作系统的过程中，将第一文件目录作为第一虚拟操作系统的根目录，其中，第一文件目录为目标操作系统

的系统内核预先为第一虚拟操作系统分配的，第一虚拟操作系统的根目录用于保存第一虚拟操作系统运行所需的文件；在第一资源上运行第一进程，第一进程为第一虚拟操作系统的系统进程，第一进程用于对第一文件目录下的文件进行处理，第一进程不同于第二进程，第二进程为第二虚拟操作系统的系统进程，在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统。

[0092] LXC操作系统容器包括1个名为“rootfs”的文件夹(如第一文件目录)和一个与LXC相关的config配置文件，也即对于系统容器，其并不是提供一个完整的计算机系统，相当于提供了一个应用运行环境(rootfs目录)，并提供一系列的运行机制(通过config配置文件实现)，在于主机共用驱动和内核的基础上，各自使用独立的硬件资源，并独立运行各自的进程，进程之间互不干扰。

[0093] 具体而言，在修改过的Host Android主机上修改Android使用的内核版本，使用Linux内核的CGroups和NameSpace功能来支持硬件的虚拟化和软件资源的虚拟化。使用NameSpace和PID完成软件资源的隔离，并使用LXC工具(LXC Tools)为每个容器启动一个启动进程并创建NameSpace(即命名空间)和对应的CGroup(分组化管理的进程)。

[0094] 可选地，使用预先修改制作好的不同操作系统的容器文件，存放在修改后Host的存储空间中，然后使用虚拟化技术来实现启动多个虚拟操作系统的目的，用户可以在多个系统中切换，并且独立的运行不同的或者相同的系统隔离的应用程序。

[0095] 对于每一个系统容器，均可以安装应用程序，系统容器之间的应用程序由于处于不同的根目录，且处理应用程序的数据的进程、CPU、内存资源不同，所以不会造成应用程序的数据泄露，保证了应用程序的数据安全。

[0096] 在该实施例中，可针对android设备(也即目标设备)，通过修改android系统内核，android系统源码，交叉编译和移植LXC，修改android设备相关驱动，以实现在一台android设备上虚拟出相同或者不同的一个或者多个操作系统容器的目的。例如，在android设备中运行不同版本的2个或者多个相同或者不同版本的android系统容器，或者在android设备中运行busybox，或者是ubuntu的系统容器。在启动时，所有容器均使用主机的驱动和硬件资源，不需要进行硬件资源的模拟，也无需加载相应的硬件驱动，从而可以提高启动的效率。

[0097] 在上述实施例中，仅以目标设备为安卓设备为例进行了说明，实际上目标设备还可以为台式机、服务器等计算机设备，作为一种可选的实施例，下面以Linux服务器为例进行说明：

[0098] 步骤S11，添加LXC工具。

[0099] 根据Linux服务器中Linux内核的版本选择对应版本的LXC工具，并将LXC工具通过交叉编译的方式添加到Linux服务器中。

[0100] 步骤S12，增加命令集合。

[0101] 根据LXC工具的版本选择对应版本的命令集合(如busybox)，然后使用adb工具将其安装到服务器上。

[0102] 步骤S13，提取系统容器。

[0103] 为每个容器定义其根目录和对应的配置文件，从而生成对应的系统容器，为了使用主机服务器的Linux内核，该系统容器中的虚拟操作系统需要支持该Linux内核，虚拟操

作系统可以为Android、Linux、Ubuntu等。

[0104] 步骤S14，运行系统容器。

[0105] 用户可以通过Linux服务器申请属于自己的系统容器，并在远端通过自己的电脑登录并使用这个系统容器，通过输入用户名和密码等信息进入到该系统中，并安装应用程序，安装的应用程序所有文件会被保存在该虚拟操作系统的根目录中，避免被其余用户或者操作系统利用。

[0106] 需要说明的是，Linux操作系统容器包括1个名为制定的文件夹(对应于前文中“rootfs”)和一个与LXC相关的config配置文件，也即对于系统容器，其相当于提供了一个应用运行目录环境，并提供一系列的运行机制(通过config配置文件实现)，在于主机共用驱动和内核的基础上，各自使用独立的硬件资源，并独立运行各自的进程，进程之间互不干扰。从而可以使得各个虚拟机之间相互隔离，互不影响。

[0107] 可选地，对于前述的LXC工具，还可以使用Docker工具来替代，实现系统中的操作系统的虚拟。

[0108] 在本申请的技术方案中，不用虚拟化硬件设备和相关驱动模拟，大幅度提升虚拟系统的启动速度。使用NameSpace和PID完成隔离，可以使用挂载功能共享Host Android的相关文件和软件资源，同时可以直接使用Host Android内核自带的相关硬件驱动，降低了虚拟系统的运行消耗。

[0109] 需要说明的是，对于前述的各方法实施例，为了简单描述，故将其都表述为一系列的动作组合，但是本领域技术人员应该知悉，本发明并不受所描述的动作顺序的限制，因为依据本发明，某些步骤可以采用其他顺序或者同时进行。其次，本领域技术人员也应该知悉，说明书中所描述的实施例均属于优选实施例，所涉及的动作和模块并不一定是本发明所必须的。

[0110] 通过以上的实施方式的描述，本领域的技术人员可以清楚地了解到根据上述实施例的方法可借助软件加必需的通用硬件平台的方式来实现，当然也可以通过硬件，但很多情况下前者是更佳的实施方式。基于这样的理解，本发明的技术方案本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来，该计算机软件产品存储在一个存储介质(如ROM/RAM、磁碟、光盘)中，包括若干指令用以使得一台终端设备(可以是手机，计算机，服务器，或者网络设备等)执行本发明各个实施例所述的方法。

[0111] 实施例2

[0112] 根据本发明实施例，还提供了一种用于实施上述虚拟操作系统的运行方法的虚拟操作系统的运行装置。图6是根据本发明实施例的一种可选的虚拟操作系统的运行装置的示意图，如图6所示，该装置可以包括：接收单元62、响应单元64以及显示单元66。

[0113] 接收单元62，用于在目标设备上接收到第一指令，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统。

[0114] 上述的目标设备运行以Linux内核(或Unix内核)为内核的系统(也即目标操作系统)，所运行的系统包括但不局限于Linux、Ubuntu、Android，该目标设备的类型包括但不局限于移动设备、计算机终端PC、服务器；目标操作系统中安装有一个或多个系统容器，每个系统容器用于提供一个虚拟操作系统，任意两个系统容器所提供的虚拟操作系统可以相同或者不同。

[0115] 在目标设备为移动设备或PC时，通过在移动设备或PC中提供系统容器(相当于虚拟操作系统)，方便用户以安全模式使用移动设备或PC，即每个用户仅能进入相应的系统容器所提供的虚拟操作系统，且多个用户所使用的系统容器相互独立、物理隔离，不会相互影响，可以提高移动设备或PC中信息的安全性。

[0116] 在目标设备为服务器时，服务器可以通过云服务的形式提供若干个系统容器，每个用户可通过互联网使用自己的系统容器，一方面可以降低用户本地终端成本(仅需一个联网设备即能进入到自己的虚拟系统)，而不用频繁升级设备；另一方面可以提高信息的安全性(由于所使用的系统容器相互独立、物理隔离)。

[0117] 响应单元64，用于响应于第一指令，获取为第一虚拟操作系统分配的第一资源，其中，第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源。

[0118] 相关技术中，对于不同的虚拟操作系统，存在共用CPU、内存等物理资源的情况，从而存在其中一个虚拟操作系统运行时的信息被其它操作系统知晓(如存在于共享内存中的信息)的可能性，从而影响了虚拟操作系统的安全性。

[0119] 对于本申请的任意两个系统容器(相当于两个虚拟操作系统)，所分配的用于运行或承载系统容器的物理资源相互独立隔离，如所使用的CPU资源、内存资源等不相同，从而可以避免虚拟操作系统运行过程中的信息被其它虚拟操作系统知晓，提高虚拟操作系统运行的安全性。

[0120] 显示单元66，用于在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面，其中，第一虚拟操作系统的系统内核为目标操作系统的系统内核，第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0121] 在相关技术中，系统启动慢，需要消耗较长时间的原因主要包括如下两个方面：(1)在启动系统前需要先用QEMU虚拟CPU等硬件设备，然后才能启动系统；(2)驱动独立且无法共享，现有系统需要虚拟出不同的硬件设备，即使虚拟出相同的硬件，每个虚拟操作系统启动时都需要运行自己的一套驱动程序。

[0122] 另外，在相关技术中，还存在技术成本高、误码率高、传输速度慢、安全性差等类似问题，对于个人用户体验不佳等等缺陷。

[0123] 对于在目标操作系统中运行的虚拟操作系统，直接利用目标操作系统中的系统内核和系统驱动进行运行，而不用为每个虚拟操作系统虚拟出物理层CPU、磁盘、图形处理设备、网络设备、内存、驱动、系统内核等，在虚拟操作系统启动时仅需建立相应的系统进程，初始化相关目录(如根目录)，并调用目标操作系统中的系统内核和系统驱动进行运行即可，而不用启动虚拟出的物理层CPU、磁盘、图形处理设备、网络设备、内存、驱动、系统内核等。从而可以解决相关技术中的问题，达到提高虚拟系统的启动效率的效果。

[0124] 需要说明的是，该实施例中的接收单元62可以用于执行本申请实施例1中的步骤S302，该实施例中的响应单元64可以用于执行本申请实施例1中的步骤S304，该实施例中的显示单元66可以用于执行本申请实施例1中的步骤S306。

[0125] 此处需要说明的是，上述模块与对应的步骤所实现的示例和应用场景相同，但不限于上述实施例1所公开的内容。需要说明的是，上述模块作为装置的一部分可以运行在如图2所示的硬件环境中，可以通过软件实现，也可以通过硬件实现。

[0126] 通过上述模块，在目标设备上接收到第一指令时，获取为第一虚拟操作系统分配的第一资源，在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统(即虚拟机)的系统界面，第一虚拟操作系统直接调用目标操作系统的系统内核，并直接调用目标操作系统的系统驱动来运行，可以解决了相关技术中虚拟机的启动较慢的技术问题，进而达到提高虚拟机的启动速度的技术效果。

[0127] 可选地，如图7所示，为了实现目标工具的交叉编译和移植，本申请的装置还可以包括：第二处理单元68，用于在目标操作系统的系统内核不包括目标工具的情况下，在接收到第二指令时，按照第二指令的指示将目标工具安装至目标操作系统的系统内核中，其中，添加至目标操作系统的系统内核中的目标工具用于为不同的系统容器分配相互隔离的物理资源。

[0128] 可选地，为了实现对目标工具的指令支持，本申请的装置还可以包括：第三处理单元，用于在目标操作系统的系统内核不包括指令集合的情况下，在接收到第三指令时，按照第三指令的指示将指令集合添加至目标操作系统中，其中，添加至目标操作系统中的指令集合用于提供运用目标操作系统中的目标工具所需的指令。

[0129] 可选地，本申请的装置还可以包括：创建单元，用于在按照第三指令的指示将指令集合添加至目标操作系统中之后，在接收到第四指令时，按照第四指令的指示运行目标工具，以创建第一系统容器和第一系统容器的配置文件，其中，指令集合包括第四指令，配置文件用于配置为第一系统容器分配的物理资源和为第一系统容器分配的文件目录。

[0130] 上述的响应单元还用于获取由目标操作系统的系统内核为第一虚拟操作系统分配的第一资源，其中，第一资源与由目标操作系统的系统内核为第二虚拟操作系统分配的第二资源不同，在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统，第二资源包括运行第二虚拟操作系统所使用的目标设备上的物理资源。

[0131] 在一个可选的实施例中，本申请的装置还可以包括：第一处理单元，用于在使用第一资源运行第一虚拟操作系统的过程中，将第一文件目录作为第一虚拟操作系统的根目录，其中，第一文件目录为目标操作系统的系统内核预先为第一虚拟操作系统分配的，第一虚拟操作系统的根目录用于保存第一虚拟操作系统运行所需的文件；运行单元，用于在第一资源上运行第一进程，其中，第一进程为第一虚拟操作系统的系统进程，第一进程用于对第一文件目录下的文件进行处理，第一进程不同于第二进程，第二进程为第二虚拟操作系统的系统进程，在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统。

[0132] 可选地，使用预先修改制作好的不同操作系统的容器文件，存放在修改后Host的存储空间中，然后使用虚拟化技术来实现启动多个虚拟操作系统的目的，用户可以在多个系统中切换，并且独立的运行不同的或者相同的系统隔离的应用程序。

[0133] 在该实施例中，可针对android设备(也即目标设备)，通过修改android系统内核，android系统源码，交叉编译和移植LXC，修改android设备相关驱动，以实现在一台android设备上虚拟出相同或者不同的一个或者多个操作系统容器的目的。例如，在android设备中运行不同版本的2个或者多个相同或者不同版本的android系统容器，或者在android设备中运行busybox，或者是ubuntu的系统容器。在启动时，所有容器均使用主机的驱动和硬件资源，不需要进行硬件资源的模拟，也无需加载相应的硬件驱动，从而可以提高启动的效

率。

[0134] 在上述实施例中，仅以目标设备为安卓设备为例进行了说明，实际上目标设备还可以为台式机、服务器等计算机设备。

[0135] 在上述装置实施例中，所涉及的单元或模块与前述方法实施例中的方法步骤相对应，对于装置实施例中的单元或模块的具体实现方式，可参考前述方法实施例中对应的方法步骤的具体实施方式。在此不再赘述。

[0136] 此处需要说明的是，上述模块与对应的步骤所实现的示例和应用场景相同，但不限于上述实施例1所公开的内容。需要说明的是，上述模块作为装置的一部分可以运行在如图2所示的硬件环境中，可以通过软件实现，也可以通过硬件实现，其中，硬件环境包括网络环境。

[0137] 实施例3

[0138] 根据本发明实施例，还提供了一种用于实施上述虚拟操作系统的运行方法的服务器或终端(也即前述的电子装置)。

[0139] 图8是根据本发明实施例的一种终端的结构框图，如图8所示，该终端可以包括：一个或多个(图8中仅示出一个)处理器801、存储器803、以及传输装置805(如上述实施例中的发送装置)，如图8所示，该终端还可以包括输入输出设备807。

[0140] 其中，存储器803可用于存储软件程序以及模块，如本发明实施例中的虚拟操作系统的运行方法和装置对应的程序指令/模块，处理器801通过运行存储在存储器803内的软件程序以及模块，从而执行各种功能应用以及数据处理，即实现上述的虚拟操作系统的运行方法。存储器803可包括高速随机存储器，还可以包括非易失性存储器，如一个或者多个磁性存储装置、闪存、或者其他非易失性固态存储器。在一些实例中，存储器803可进一步包括相对于处理器801远程设置的存储器，这些远程存储器可以通过网络连接至终端。上述网络的实例包括但不限于互联网、企业内部网、局域网、移动通信网及其组合。

[0141] 上述的传输装置805用于经由一个网络接收或者发送数据，还可以用于处理器与存储器之间的数据传输。上述的网络具体实例可包括有线网络及无线网络。在一个实例中，传输装置805包括一个网络适配器(Network Interface Controller,NIC)，其可通过网线与其他网络设备与路由器相连从而可与互联网或局域网进行通讯。在一个实例中，传输装置805为射频(Radio Frequency,RF)模块，其用于通过无线方式与互联网进行通讯。

[0142] 其中，具体地，存储器803用于存储应用程序。

[0143] 处理器801可以通过传输装置805调用存储器803存储的应用程序，以执行下述步骤：在目标设备上接收到第一指令，其中，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统；响应于第一指令，获取为第一虚拟操作系统分配的第一资源，其中，第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源；在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面，其中，第一虚拟操作系统的系统内核为目标操作系统的系统内核，第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0144] 处理器801还用于执行下述步骤：将第一文件目录作为第一虚拟操作系统的根目录，其中，第一文件目录为目标操作系统的系统内核预先为第一虚拟操作系统分配的，第一

虚拟操作系统的根目录用于保存第一虚拟操作系统运行所需的文件;在第一资源上运行第一进程,其中,第一进程为第一虚拟操作系统的系统进程,第一进程用于对第一文件目录下的文件进行处理,第一进程不同于第二进程,第二进程为第二虚拟操作系统的系统进程,在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统。

[0145] 采用本发明实施例,在目标设备上接收到第一指令时,获取为第一虚拟操作系统分配的第一资源,在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统(即虚拟机)的系统界面,第一虚拟操作系统直接调用目标操作系统的系统内核,并直接调用目标操作系统的系统驱动来运行,可以解决了相关技术中虚拟机的启动较慢的技术问题,进而达到提高虚拟机的启动速度的技术效果。

[0146] 可选地,本实施例中的具体示例可以参考上述实施例1和实施例2中所描述的示例,本实施例在此不再赘述。

[0147] 本领域普通技术人员可以理解,图8所示的结构仅为示意,终端可以是智能手机(如Android手机、iOS手机等)、平板电脑、掌上电脑以及移动互联网设备(Mobile Internet Devices,MID)、PAD等终端设备。图8其并不对上述电子装置的结构造成限定。例如,终端还可包括比图8中所示更多或者更少的组件(如网络接口、显示装置等),或者具有与图8所示不同的配置。

[0148] 本领域普通技术人员可以理解上述实施例的各种方法中的全部或部分步骤是可以通过程序来指令终端设备相关的硬件来完成,该程序可以存储于一计算机可读存储介质中,存储介质可以包括:闪存盘、只读存储器(Read-Only Memory,ROM)、随机存取器(Random Access Memory,RAM)、磁盘或光盘等。

[0149] 实施例4

[0150] 本发明的实施例还提供了一种存储介质。可选地,在本实施例中,上述存储介质可以用于执行虚拟操作系统的运行方法的程序代码。

[0151] 可选地,在本实施例中,上述存储介质可以位于上述实施例所示的网络中的多个网络设备中的至少一个网络设备上。

[0152] 可选地,在本实施例中,存储介质被设置为存储用于执行以下步骤的程序代码:

[0153] S21,在目标设备上接收到第一指令,其中,目标设备上运行有目标操作系统,第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统;

[0154] S22,响应于第一指令,获取为第一虚拟操作系统分配的第一资源,其中,第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源;

[0155] S23,在目标操作系统的系统界面上显示第一系统容器的目标窗口,并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面,其中,第一虚拟操作系统的系统内核为目标操作系统的系统内核,第一虚拟操作系统的系统驱动为目标操作系统的系统驱动。

[0156] 可选地,存储介质还被设置为存储用于执行以下步骤的程序代码:

[0157] S31,将第一文件目录作为第一虚拟操作系统的根目录,其中,第一文件目录为目标操作系统的系统内核预先为第一虚拟操作系统分配的,第一虚拟操作系统的根目录用于保存第一虚拟操作系统运行所需的文件;

[0158] S32,在第一资源上运行第一进程,其中,第一进程为第一虚拟操作系统的系统进程,第一进程用于对第一文件目录下的文件进行处理,第一进程不同于第二进程,第二进程为第二虚拟操作系统的系统进程,在目标操作系统中还运行有由第二系统容器提供的第二虚拟操作系统。

[0159] 可选地,本实施例中的具体示例可以参考上述实施例1和实施例2中所描述的示例,本实施例在此不再赘述。

[0160] 可选地,在本实施例中,上述存储介质可以包括但不限于:U盘、只读存储器(ROM,Read-Only Memory)、随机存取存储器(RAM,Random Access Memory)、移动硬盘、磁碟或者光盘等各种可以存储程序代码的介质。

[0161] 上述本发明实施例序号仅仅为了描述,不代表实施例的优劣。

[0162] 上述实施例中的集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时,可以存储在上述计算机可读取的存储介质中。基于这样的理解,本发明的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来,该计算机软件产品存储在存储介质中,包括若干指令用以使得一台或多台计算机设备(可为个人计算机、服务器或者网络设备等)执行本发明各个实施例所述方法的全部或部分步骤。

[0163] 在本发明的上述实施例中,对各个实施例的描述都各有侧重,某个实施例中没有详述的部分,可以参见其他实施例的相关描述。

[0164] 在本申请所提供的几个实施例中,应该理解到,所揭露的客户端,可通过其它的方式实现。其中,以上所描述的装置实施例仅仅是示意性的,例如所述单元的划分,仅仅为一种逻辑功能划分,实际实现时可以有另外的划分方式,例如多个单元或组件可以结合或者可以集成到另一个系统,或一些特征可以忽略,或不执行。另一点,所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口,单元或模块的间接耦合或通信连接,可以是电性或其它的形式。

[0165] 所述作为分离部件说明的单元可以是或者也可以不是物理上分开的,作为单元显示的部件可以是或者也可以不是物理单元,即可以位于一个地方,或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部单元来实现本实施例方案的目的。

[0166] 另外,在本发明各个实施例中的各功能单元可以集成在一个处理单元中,也可以是各个单元单独物理存在,也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现,也可以采用软件功能单元的形式实现。

[0167] 以上所述仅是本发明的优选实施方式,应当指出,对于本技术领域的普通技术人员来说,在不脱离本发明原理的前提下,还可以做出若干改进和润饰,这些改进和润饰也应视为本发明的保护范围。

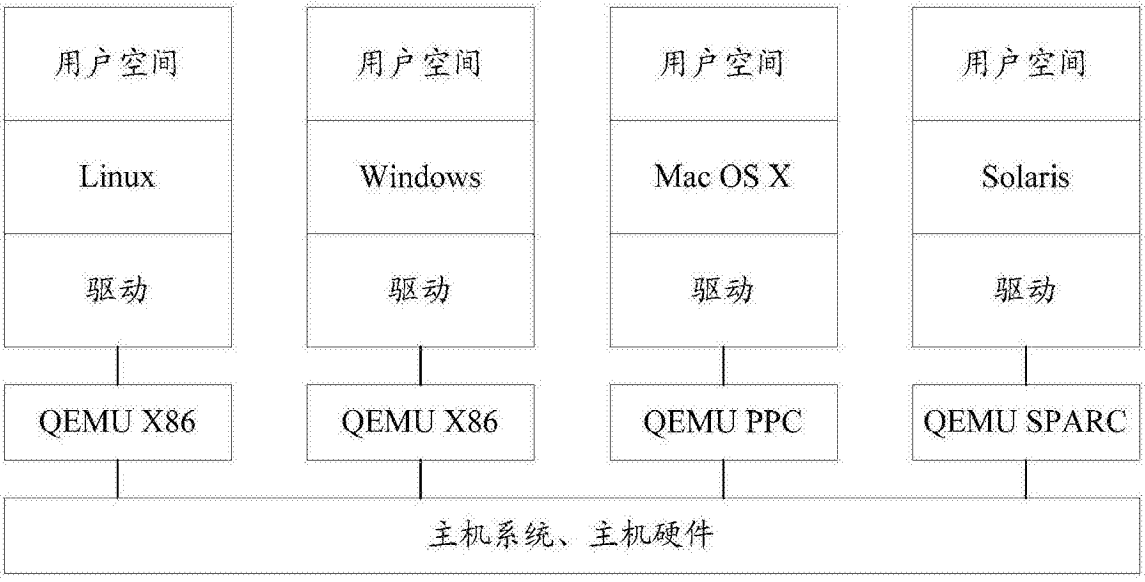

用户空间
用户空间
用户空间
用户空间
Linux
Windows
Mac OS X
Solaris
驱动
驱动
驱动
驱动
QEMU X86
QEMU X86
QEMU PPC
QEMU SPARC
主机系统、主机硬件

图1

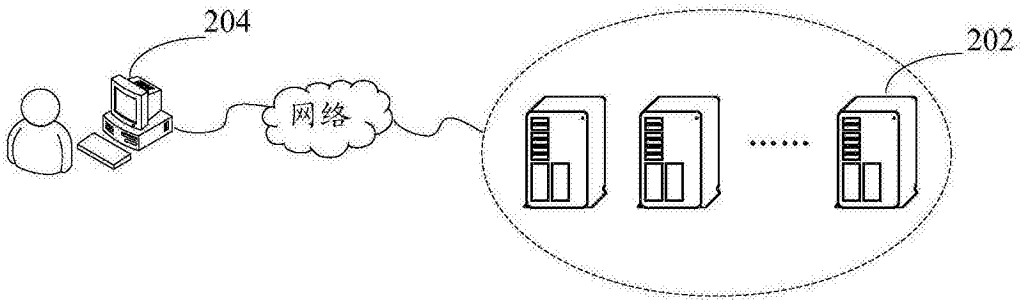

204
网络
202

图2

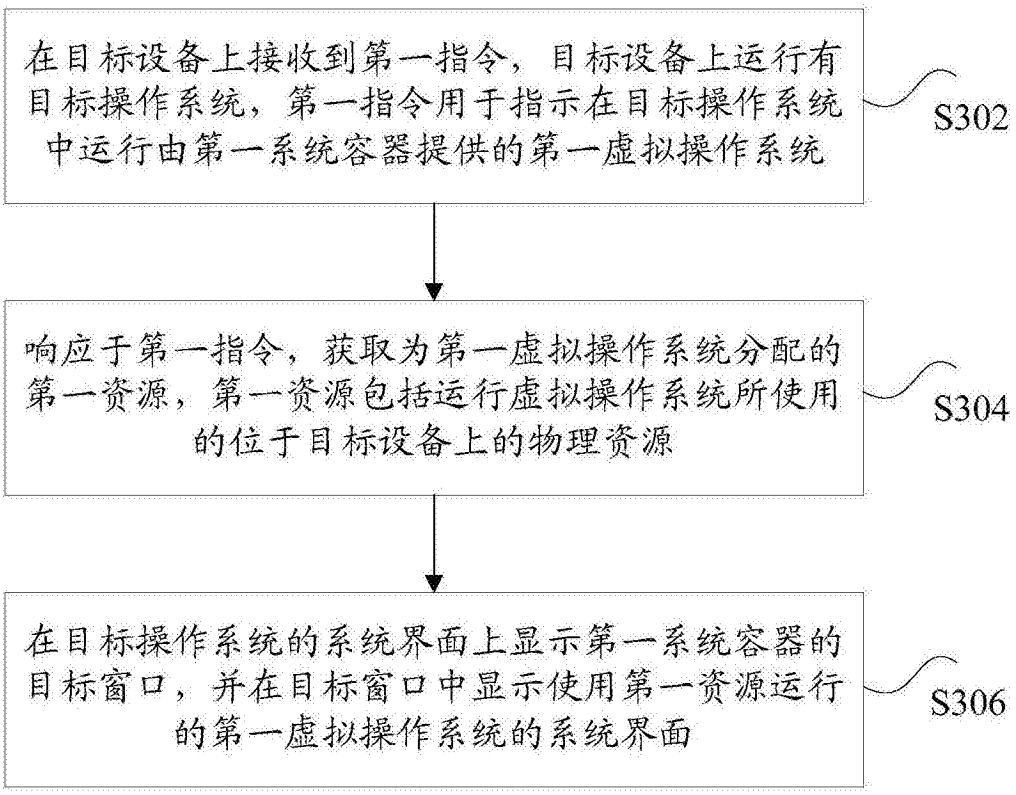

在目标设备上接收到第一指令，目标设备上运行有目标操作系统，第一指令用于指示在目标操作系统中运行由第一系统容器提供的第一虚拟操作系统
S302
响应于第一指令，获取为第一虚拟操作系统分配的第一资源，第一资源包括运行虚拟操作系统所使用的位于目标设备上的物理资源
S304
在目标操作系统的系统界面上显示第一系统容器的目标窗口，并在目标窗口中显示使用第一资源运行的第一虚拟操作系统的系统界面
S306

图3

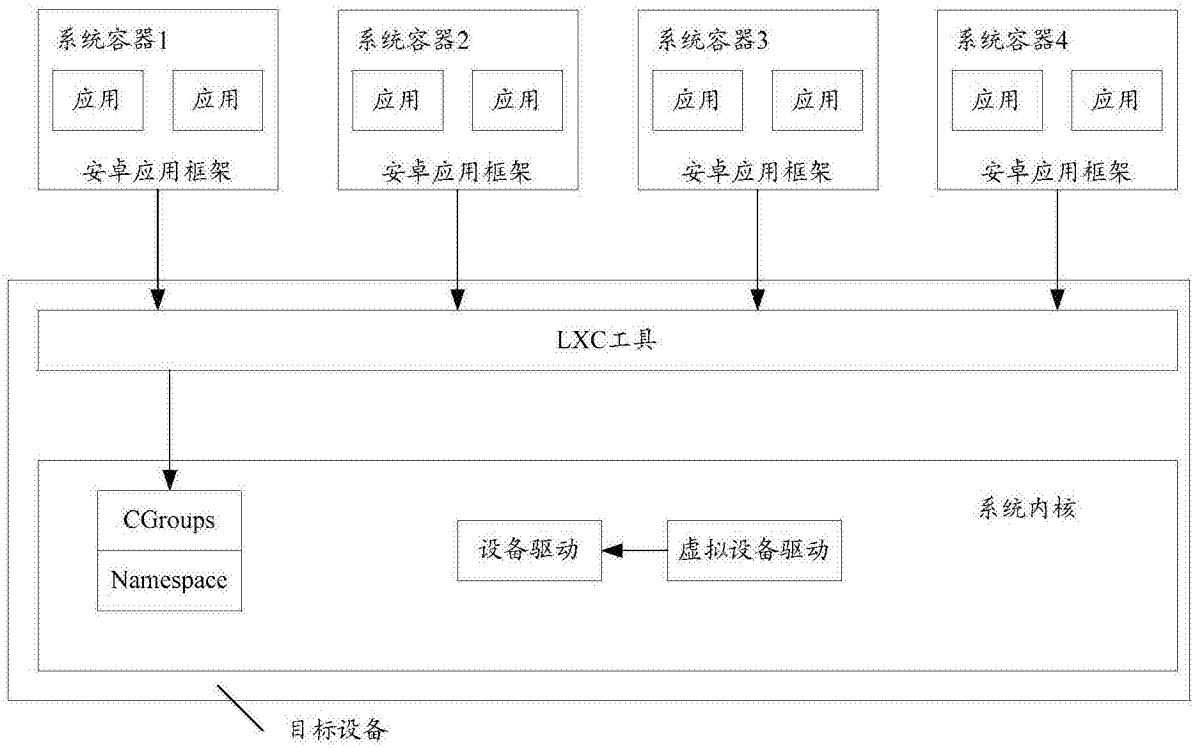

系统容器1
应用
应用
安卓应用框架
系统容器2
应用
应用
安卓应用框架
系统容器3
应用
应用
安卓应用框架
系统容器4
应用
应用
安卓应用框架
LXC工具
CGroups
Namespace
设备驱动
虚拟设备驱动
系统内核
目标设备

图4

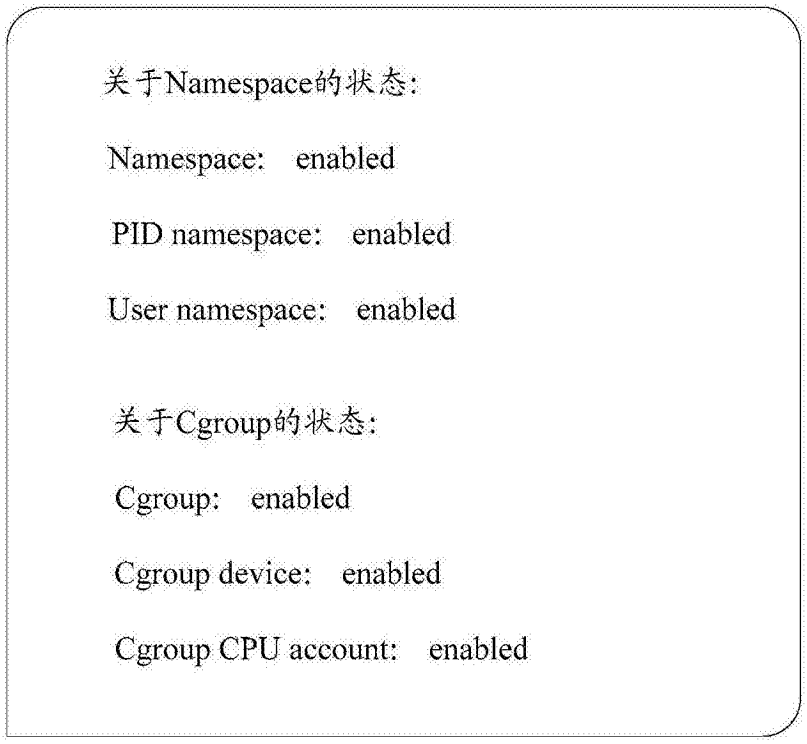

关于Namespace的状态:
Namespace: enabled
PID namespace: enabled
User namespace: enabled
关于Cgroup的状态:
Cgroup: enabled
Cgroup device: enabled
Cgroup CPU account: enabled

图5

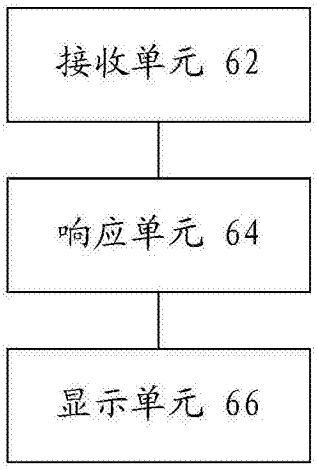

接收单元 62
响应单元 64
显示单元 66

图6

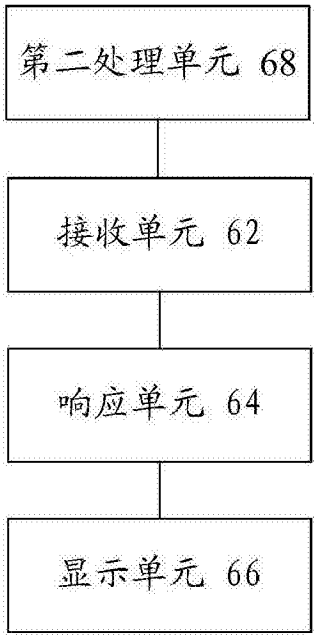


图7

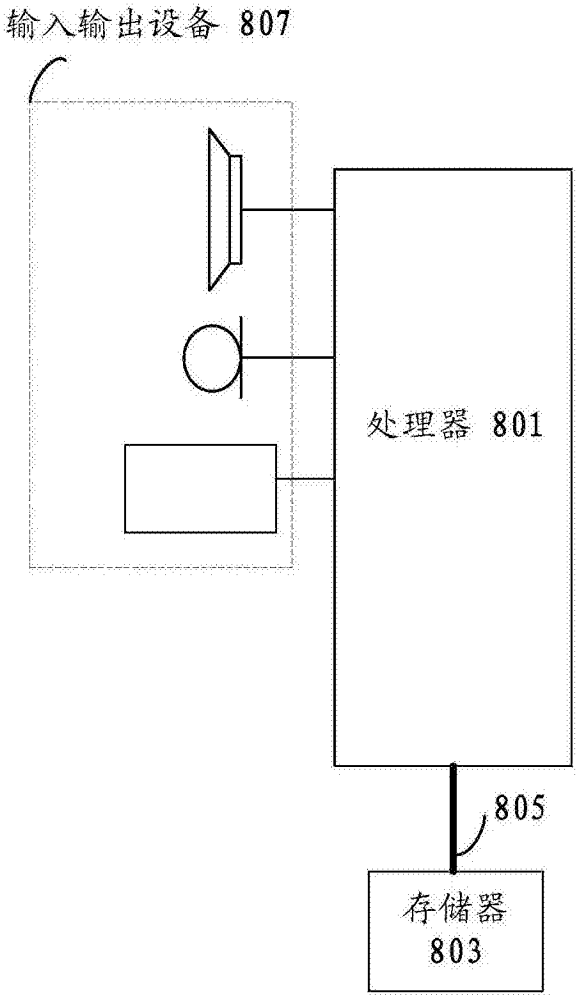


图8